2-descent for Second Order Linear Differential Equations

Tingting Fang & Mark van Hoeij*
Florida State University, Tallahassee, FL 32306-3027, USA
tfang@math.fsu.edu & hoeij@math.fsu.edu

ABSTRACT

Let L be a second order linear ordinary differential equation with coefficients in $\mathbb{C}(x)$. The goal in this paper is to reduce L to an equation that is easier to solve. The starting point is an irreducible L, of order two, and the goal is to decide if L is projectively equivalent to another equation $\tilde{L}$ that is defined over a subfield $\mathbb{C}(f)$ of $\mathbb{C}(x)$.

This paper treats the case of 2-descent, which means reduction to a subfield with index $[\mathbb{C}(x) : \mathbb{C}(f)] = 2$. Although the mathematics has already been treated in other papers, a complete implementation could not be given because it involved a step for which we do not have a complete implementation. The contribution of this paper is to give an approach that is fully implementable [5]. Examples illustrate that this algorithm is very useful for finding closed form solutions (2-descent, if it exists, reduces the number of true singularities from n to at most $n/2 + 2$).

Categories and Subject Descriptors

G.4 [**Mathematical Software**]: Algorithm design and analysis; I.1.2 [**Symbolic and Algebraic Manipulation**]: Algorithms—*Algebraic algorithms*

General Terms

Algorithms

1. INTRODUCTION

Let $L = \sum_{i=0}^{n} a_i \partial^i$ be a differential operator with coefficients in a differential field $K = \mathbb{C}(x)$, where ∂ is the usual differentiation $\frac{d}{dx}$. The corresponding differential equation is $L(y) = 0$, i.e. $a_n y^{(n)} + \cdots + a_1 y' + a_0 y = 0$. The problem of finding closed form solutions of L becomes easier if we can factor L as a product of lower order operators [2, 7, 1] or apply some other approach to reduce the order [9, 14].

A different type of reduction is called *descent*. Here, the goal is to reduce L to an operator $\tilde{L}$ of the same order, but this time defined over a proper subfield $k = \mathbb{C}(f)$ of K. Here $\tilde{L}$ must be *projectively equivalent* to L. Informally, this means that L can be solved in terms of the solutions of $\tilde{L}$ and vice versa (a precise definition will be given in Section 2.2).

In this paper, we treat the case of 2-descent, meaning that k is a subfield of K with index 2. For now, we treat only second order equations. After applying Kovacic' algorithm, we can assume that L is irreducible (i.e. not a product of lower order factors), and that it has no Liouvillian solutions.

Descent reduces the number of true singularities (Definition 5) from n to $\leq n/2+2$, which helps to solve differential equations as illustrated in Section 7. In particular, if the number of true singularities[1] drops to 3, and if these are regular singularities[2], then a ${}_2F_1$-type solution can be obtained quickly. We can also stop reducing when we reach an operator with four true singularities, because 4-singularity equations with ${}_2F_1$-type solutions are currently being classified [6] by van Hoeij and Vidunas. Classifying equations with closed form solutions and > 4 singularities would be hard to do, this is where 2-descent becomes crucial.

If $L \in \mathbb{C}(x)[\partial]$ then there is a finitely generated extension $\mathbb{Q} \subseteq C$ with $L \in C(x)[\partial]$, just take C to be the extension of $\mathbb{Q}$ given by the coefficients of L. The main design goal for our algorithm is to introduce as few algebraic extensions of C as possible. Without this design goal, Sections 3 and 5 would have been much shorter (if we simply compute the splitting field of the singularities then for Section 5 we can follow [3] and Section 3 becomes trivial. Sections 3 and 5 become non-trivial when we aim to minimize field extensions).

The main results in this paper are in Section 4. We know from [11] that if there is a gauge transformation G from L to $\sigma(L)$, then L will allow descent with respect to σ. The question is, given G, how to find the descent? Is it necessary (as in the terminology in [11]) to trivialize a 2-cocycle, or to perform some equivalent complicated operation such as finding a point on a conic over $C(x)$? The answer is no; we give a short and efficient algorithm in Section 4, and we even show (Theorem 1) that it produces a result over an optimal extension of C.

1.1 Relation to prior work

It is shown in [3, 11] that the problem of computing 2-descent can be reduced to another problem (trivializing a 2-cocycle) although no step by step algorithm is given in these papers. The paper [9] does give an algorithm, and im-

*Supported by NSF grant 1017880.

ISSAC'11, June 8–11, 2011, San Jose, California, USA.

[1] the number of *removable* singularities (Def. 5) is irrelevant
[2] for the irregular singular case, finding closed form solutions if they exist can be done with [12, 4]

plementation, that can be used to find 2-descent, as follows. If σ is a Möbius transformation of order 2, and $\mathbb{C}(f)$ is the fixed field of σ, and if L is projectively equivalent to $\sigma(L)$, then we can compute the so-called symmetric product of $L, \sigma(L)$, then apply factorization (DFactorLCLM in Maple), take the 3'rd order factor found that way, and run the algorithm from [9] to find a second order operator. All of these steps are implemented, and the end result is a 2-descent.

The problem with the above methods is that they rely on an algorithm that can find a point on a conic defined over K (or an algorithm that solves an equivalent problem). Although such a point must exist when $K = \mathbb{C}(x)$, the proof does not show how to find such a point over a field of constants that is optimal or close to optimal (recall that we wish to minimize the extension of C that the algorithm introduces, where $C \subset \mathbb{C}$). There is only an implementation [10] for this step if C is $\mathbb{Q}$ or a transcendental extension of $\mathbb{Q}$. If L contains algebraic numbers, then there is no implementation for finding a point on a conic, and without that, it is not clear how to obtain from [11, 9, 3] a complete implementation for finding 2-descent.

In this paper we describe a step by step algorithm for finding 2-descent. The algorithm can be fully implemented [5] because it does not call a conic algorithm. Note: If $L \in C(x)[\partial]$ with $C \subset \mathbb{C}$, and if one allows unnecessary algebraic extensions of C (potentially exponentially large), then it is not hard to implement a conic algorithm, in which case one can consider 2-descent an already solved problem. But in practice our algorithm would be much preferable because it only extends C when necessary (i.e. when there is no 2-descent defined over C).

2. PRELIMINARIES

2.1 Differential Operators and Singularities

Let $K = \mathbb{C}(x)$ denote the differential field and let $\mathcal{D}=K[\partial]$ be the ring of differential operators with coefficients in the differential field K. Here ∂ denotes the usual differentiation $\frac{d}{dx}$. Then elements $L \in \mathcal{D}$ are of the form $L = a_n\partial^n + \cdots + a_1\partial + a_0$ with $a_i \in K$.

A point $p \in \mathbb{P}^1 = \mathbb{C} \cup \{\infty\}$ is called a *singularity* of a differential operator $L \in K[\partial]$, if p is a zero of the leading coefficient of L or p is a pole of one of the other coefficients of L. p is called a *regular* point if it is not a *singularity*.

We denote the solution space of a differential operator as $V(L) = \{y|L(y) = 0\}$ where the y are taken in some universal extension [15] of $\mathbb{C}(x)$. If p is a regular point of L, we can write all solutions of L at p as convergent power series $\sum_{i=0}^{\infty} a_i t_p^i$, where t_p denotes the local parameter which is $t_p = \frac{1}{x}$ if $p = \infty$ and $t_p = x - p$, otherwise.

2.2 Transformations

There are three known types of transformations that send, for any second order $L_1 \in K[\partial]$, the solution space of L_1 to the solution space of some $L_2 \in K[\partial]$, again of order 2. They are (notation as in [4]):

(i) change of variables: $y(x) \to y(f(x))$, $\quad f(x) \in K \setminus \mathbb{C}$.

(ii) exp-product: $y \to e^{\int r\,dx} \cdot y$, $\quad r \in K$.

(iii) gauge transformation: $y \to r_0 y + r_1 y'$, $\quad r_0, r_1 \in K$.

Definition 1. *Let $L_1, L_2 \in K[\partial]$ with order 2. They are called gauge equivalent (notation: $L_1 \sim_g L_2$) if there exists a so-called gauge transformation from $V(L_1)$ to $V(L_2)$, which means a bijection of the form (iii).*

Remark 1. *Let L_1, $L_2 \in K[\partial]$. The $\mathcal{D}$-modules $\mathcal{D}/\mathcal{D}L_i$, $i = 1, 2$ are isomorphic if and only if $L_1 \sim_g L_2$. In particular, $\sim_g$ is an equivalence relation (see [1]).*

Definition 2. *Let L_1, $L_2 \in K[\partial]$ with order 2. They are called* projectively equivalent *(notation: $L_1 \sim_p L_2$) if there exists a bijection $V(L_1) \to V(L_2)$ of the form*

$$y \longrightarrow e^{\int r} \cdot (r_0 y + r_1 y') \tag{1}$$

for some r, r_0, $r_1 \in K$.

Projective equivalence is also an equivalence relation, see [1]. An implementation (for order 2) is given in [8] to decide if $L_1 \sim_p L_2$, and if so, to find the projective equivalence (the r, r_0, r_1 in (1)). An algorithm for arbitrary order n was given in [1] (implemented in ISOLDE).

2.3 2-descent

Definition 3. *Let $f = \frac{A}{B}$ with $A, B \in \mathbb{C}[x]$ coprime, then the degree of f is defined as*

$$\deg(f) = \max(\deg(A), \deg(B)) = [\mathbb{C}(x) : \mathbb{C}(f)].$$

Remark 2. *If $\sigma \in \mathrm{Aut}(\mathbb{C}(x)/\mathbb{C})$ has order 2, then the fixed field of σ is a subfield of $\mathbb{C}(x)$ of index 2, and by Lüroth's theorem this subfield is of the form $\mathbb{C}(f)$, for some $f \in \mathbb{C}(x)$ of degree 2 (note: we can find such f in $\{x + \sigma(x), x\sigma(x)\} \setminus C$). Any subfield $\mathbb{C}(f) \subset \mathbb{C}(x)$ of index 2 is the fixed field of some $\sigma \in \mathrm{Aut}(\mathbb{C}(x)/\mathbb{C})$ of order 2 (after all, every extension of degree 2 is Galois). The automorphisms of $\mathbb{C}(x)$ over $\mathbb{C}$ are Möbius transformations:*

$$x \mapsto \frac{ax + b}{cx + d} \tag{2}$$

This paper treats 2-descent, so we only consider σ of order 2, which is equivalent to having $d = -a$ in (2).

Remark 3. *Any $\sigma \in \mathrm{Aut}(\mathbb{C}(x)/\mathbb{C})$ extends to an automorphism of $\mathbb{C}(x)[\partial]$. If σ has finite order, and if $\mathbb{C}(f)$ is the fixed field of σ, and if $L \in \mathbb{C}(x)[\partial]$, then*

$$L = \sigma(L) \Longleftrightarrow L \in \mathbb{C}(f)[\partial_f], \tag{3}$$

in other words, $\mathbb{C}(f)[\partial_f]$ is the fixed ring of σ. Here $\partial_f := \frac{d}{df} = \frac{1}{f'}\partial$, where $'$ is differentiation w.r.t. x.

Definition 4. *Let $L \in \mathbb{C}(x)[\partial]$. We say that L has 2-descent if $\exists f \in \mathbb{C}(x)$ with $\deg(f) = 2$ and $\exists \tilde{L} \in \mathbb{C}(f)[\partial_f]$ such that $L \sim_p \tilde{L}$.*

One could instead use the term "projective 2-descent" for this (because we use projective equivalence $\sim_p$) but we opted to use the shorter term.

Main goal: Let $L \in K[\partial]$ be irreducible and of order 2. The goal of this paper is to give an explicit algorithm that can decide if L has 2-descent, and if so, find it (i.e. find $\tilde{L} \in \mathbb{C}(f)[\partial_f]$ with $L \sim_p \tilde{L}$ for some f of degree 2). Moreover, if L is defined over some field $C \subset \mathbb{C}$, we should only introduce algebraic extensions of C when necessary.

We will divide our algorithm into several steps. The first step is to find candidates for $\mathbb{C}(f)$ with $\deg(f) = 2$. Such a field is the fixed field of a Möbius transformation of order 2.

3. MÖBIUS TRANSFORMATIONS

PROPOSITION 1. *A Möbius transformation has order 2 if it is of the form* $\sigma(x) = \frac{ax+b}{cx-a}$. *Such* σ *has 2 fixed points in* $\mathbb{C} \cup \{\infty\}$.

One could apply a transformation that moves the fixed points of σ to $0, \infty$, which reduces σ to the notationally convenient $x \mapsto -x$. Our algorithm does not do this because it can introduce an unnecessary algebraic extension of the constants.

3.1 The singularity structure

DEFINITION 5. *Let* $L \in \mathcal{D}$ *have order* n. *Assume* p *is a singularity of* L. *If there exists a basis of* $V(L)$ *of the form* $e^{\int r} f_1, \ldots, e^{\int r} f_n$ *where* $r \in \mathbb{C}(x)$ *and* $f_1, \ldots, f_n$ *are analytic at* $x = p$, *then* p *is called a* removable *singularity (also called* false *singularity). Otherwise* p *is called a* true *singularity.*

Suppose p is a singularity of L. If there exists a projectively equivalent $\tilde{L}$ for which p is a regular point, then p is a removable singularity. The true singularities of L are precisely those p that stay singular when L is replaced by any projectively equivalent operator.

Denote (as in [12, 4]) the (generalized) exponent-difference as $\Delta(L, p)$.

DEFINITION 6. *For any true singularity* p, *denote*

$$\text{type}(L,p) := \begin{cases} \text{"irreg"} & \text{if} \quad \Delta(L,p) \notin \mathbb{C} \\ \text{"irrat"} & \text{if} \quad \Delta(L,p) \in \mathbb{C} \setminus \mathbb{Q} \\ e \in [0, \frac{1}{2}] & \text{if} \quad \Delta(L,p) \in \mathbb{Q} \end{cases}$$

Here, $e \in [0, \frac{1}{2}]$ such that $\Delta(L, p) \in (e + \mathbb{Z}) \cup (-e + \mathbb{Z})$. Then we write the *singularity structure* of L as

$$S^{\text{type}} := \{(p, \text{type}(L, p)) \mid p \text{ true sing}\}.$$

Let π_i project on the i'th entry of S^{type}, then $S := \pi_1(S^{\text{type}}) \subseteq \mathbb{P}^1(\mathbb{C})$ denotes the set of true singularities of L.

LEMMA 1. *[12, 4] If* $L \sim_p \tilde{L} \in \mathcal{D}$ *then* L *and* $\tilde{L}$ *have the same singularity structure* S^{type}.

If $L \in C(x)[\partial]$ for some field $C \subset \mathbb{C}$, we denote:

$$M_{\mathbb{C}} := \{\sigma = \frac{ax+b}{cx-a} \mid a, b, c \in \mathbb{C} \text{ and } \sigma(S) = S\}$$

$$M_C := \{\sigma = \frac{ax+b}{cx-a} \mid a, b, c \in C \text{ and } \sigma(S) = S\}$$

$$M_{\mathbb{C}}^{\text{type}} := \{\sigma \in M_{\mathbb{C}} \mid \sigma(S^{\text{type}}) = S^{\text{type}}\}$$

$$M_C^{\text{type}} := \{\sigma \in M_C \mid \sigma(S^{\text{type}}) = S^{\text{type}}\}$$

$$\text{places}(C) := \{f \in C[x] \mid f \text{ is monic and irreducible }\} \bigcup \{\infty\}.$$

REMARK 4. $\text{places}(\mathbb{C}) \cong \mathbb{P}^1(\mathbb{C}) = \mathbb{C} \bigcup \{\infty\}$

If $\sigma \in \text{Aut}(C(x)/C)$ then σ acts on places(C) in a natural way, preserving degrees, which are defined as:

$$\deg(p) = \begin{cases} 1 & \text{if} \quad p = \infty; \\ \deg(p) & \text{if} \quad p \text{ is a polynomial} . \end{cases}$$

If $L = a_n \partial^n + \cdots + a_0 \partial^0$ with $a_0, \ldots, a_n \in C[x]$, then computing the singularities as a subset of $\mathbb{P}^1(\overline{C}) \subset \mathbb{P}^1(\mathbb{C})$ would mean computing all roots (the splitting field) of a_n. The algorithm does not compute this splitting field because it could have exponentially high degree over C. Instead, it uses irreducible factors of a_n in $C[x]$ (and the point ∞) to represent the singularities, then we have the notation S_C^{type} and

$$M_C^{\text{type}} := \{\sigma \in M_C \mid \sigma(S_C^{\text{type}}) = S_C^{\text{type}}\}$$

To ensure that S is invariant under $\sim_p$ it is essential to discard all removable singularities.

EXAMPLE 1. *Let* $C = \mathbb{Q}$, *and*

$$L := \partial^2 + \frac{12x^4 + 1}{x(2x^2 - 1)(2x^2 + 1)} \partial - \frac{8}{(2x^2 - 1)^2}$$

For this example we find

$$S^{\text{type}} := \{(\infty, 0), (0, 0), (\frac{-1}{\sqrt{2}}, 0), (\frac{1}{\sqrt{2}}, 0), (\frac{-1}{\sqrt{-2}}, 0), (\frac{1}{\sqrt{-2}}, 0)\}.$$

The set of true singularities is

$$S = \pi_1(S^{\text{type}}) = \{\infty, 0, \frac{1}{\sqrt{2}}, \frac{-1}{\sqrt{2}}, \frac{1}{\sqrt{-2}}, \frac{-1}{\sqrt{-2}}\}$$

Written in terms of places$(\mathbb{Q})$ *it becomes*

$$S_C := \{\infty, x, x^2 + \frac{1}{2}, x^2 - \frac{1}{2}\} \subset \text{places}(\mathbb{Q}),$$

$$S_C^{\text{type}} := \{(\infty, 0), (x, 0), (x^2 + \frac{1}{2}, 0), (x^2 - \frac{1}{2}, 0)\}$$

and

$$M_C^{\text{type}} = \{-x, \frac{1}{2x}, \frac{-1}{2x}\}.$$

This example was quite easy because it has obvious 2-descent. Moreover, all singularities were true singularities with $\text{type}(L, p) = 0$. Removable singularities are common in larger examples, such as Example 3 in Section 7. Using S instead of S_C would have introduced an extension of $C = \mathbb{Q}$ of degree 4 in this example, however, such an extension could have been much larger (e.g. if $x^5 - x - 1$ had appeared in the denominator of L, which has a splitting field of degree 120).

3.2 Finding candidates for σ

For $i = 1, 2, \ldots$, let S_i denote the set of all $p \in S_C$ with $\deg(p) = i$.

Algorithm: Compute Möbius transformations.

Input: The singularity structure S_C^{type}.

Output: The set M_C^{type}, i.e., the set of all $\sigma \in \text{Aut}(C(x)/C)$ of order 2 that fix S_C^{type}. (In this paper we omit 2-descent for σ's that are not defined over C because in that case is better to compute a larger descent, of type $C_2 \times C_2$, D_n, A_4, S_4, or A_5).

Step 1: Compute S_i from S_C^{type} and let n_i denote the number of elements of S_i.

Step 2: Let $n_{sing} := \sum i\, n_i$ (the total number of true singularities when counted in $\mathbb{P}^1(\overline{C})$).

Step 3: If $n_{sing} < 3$ then return "With < 3 singularities, descent is not necessary nor implemented" and stop.

Step 4: Now $n_{sing} \geq 3$.

(i) If $n_1 \geq 3$, then call **Case1**
(ii) If $n_1 = 1, n_2 = 1$, then call **Case2**
(iii) If $n_1 = 2, n_2 = 1$, then call **Case3**
(iv) If $n_2 \geq 2$, then call **Case4**
(v) If $n_i \geq 1$ for some $i \geq 3$, then call **Case5**

Algorithm: Case1.

Input: S_C^{type} with S_1 having ≥ 3 elements.

Output: The set M_C^{type}.

Before describing Algorithm Case1, first some remarks. In general $\sigma = \frac{ax+b}{cx+d}$ is determined by the image of three points $\sigma(p_1), \sigma(p_2), \sigma(p_3)$. Since we assume $|\sigma| = 2$, we can write $\sigma = \frac{ax+b}{cx-a}$. In general, such σ is determined by two points $\sigma(p_1)$, $\sigma(p_2)$ except in one case: when $\sigma(p_1) = p_2$, $\sigma(p_2) = p_1$. In that case one more point is needed to determine $\sigma = \frac{ax+b}{cx-a}$.

Algorithm Case1 will choose a pair $p_1, p_2 \in S_1$ ($p_1 \neq p_2$) and loops over all $n(n-1)$ pairs $q_1, q_2 \in S_1$ ($q_1 \neq q_2$). If the types of q_1, q_2 match those of p_1, p_2, the algorithm will compute the σ that maps p_1, p_2 to q_1, q_2. In the one case that $q_1, q_2 = p_2, p_1$, a third point p_3 is used to determine σ. There are $n-2$ choices for $\sigma(p_3)$, namely from $S_1 - \{p_1, p_2\}$. The number of computed σ's is then $\leq n(n-1) - 1 + (n-2)$ (equality if they all have the same type). Then we remove those σ for which S_C^{type} is not σ-invariant (That means remove all σ's that send a true singularity to a non-singular point or to a false singularity (Definition 5), and, remove all σ's that send a singularity to a singularity of a different type).

Algorithm: Case2

Input: S_C^{type} with S_1 having 1 element and S_2 having 1 element.

Output: The set M_C^{type}.

Step 1: Let the polynomial in S_2 be $x^2 + c_1 x + c_0$.

Step 2: Write $\sigma_1 = -\frac{c_1\,x + 2c_0}{2x + c_1}$ and $\sigma_2 = \frac{a\,x + c_0\,c + c_1\,a}{c\,x - a}$.

REMARK 5. *σ_1 is the unique Möbius transformation of order 2 that fixes the roots of $x^2 + c_1 x + c_0$; σ_2 is the parameterized family of all σ of order 2 that swap the roots of $x^2 + c_1 x + c_0$.*

Step 3: Let p_1 be the one element of S_1. Equating $\sigma(p_1)$ to p_1 gives a linear equation that determines the values of the homogeneous parameters a, c in σ_2.

Step 4: Check which (if any) of σ_1, σ_2 fix S_C^{type} and return those.

Algorithm **Case3** is similar to Algorithm **Case2**.

Algorithm: Case4

Input: S_C^{type} with S_2 having ≥ 2 elements.

Output: The set M_C^{type}.

Step 1: Choose one polynomial from S_2. Denote it as $f_1 = x^2 + c_1\,x + c_0$.

Step 2: Do the following substeps $1 - 4$ to get the set T_1:

1. Write $\sigma_1 = -\frac{c_1\,x + 2c_0}{2x + c_1}$ and $\sigma_2 = \frac{a\,x + c_0\,c + c_1\,a}{c\,x - a}$ (See the Remark in Algorithm Case2).
2. Choose another polynomial in S_2, and denote it as $f_2 = x^2 + d_1\,x + d_0$.
3. Write $\sigma_3 = -\frac{d_1\,x + 2d_0}{2x + d_1}$ and $\sigma_4 = \frac{a\,x + d_0\,c + d_1\,a}{c\,x - a}$.
4. Let $a := d_0 - c_0$, $c := c_1 - d_1$, then $\sigma_2 = \sigma_4$ swaps the roots of f_1 as well as the roots of f_2.
   $T_1 := \{\sigma \in \{\sigma_1, \sigma_2, \sigma_3\} | \sigma \text{ fixes } S_C^{\text{type}}\}$.

Step 3: Denote the polynomials in S_2 as f_i, then $T_2 := \bigcup_{i=2}^{n_2} \text{FindMaps}(f_1, f_i)$
(See below for the subalgorithm **FindMaps**)

Step 4: $T_3 := \bigcup_{i=3}^{n_2} \text{FindMaps}(f_2, f_i)$.

Step 5: $T_1 \bigcup T_2 \bigcup T_3$.
Remark. Taking a set union means removing duplicates. The duplicates are the elements of T_3 that do not swap the roots of f_1, and σ_3 might also be duplicate (it could be in T_2 if $n_2 > 2$).

Subalgorithm: FindMaps

Input: Two irreducible polynomials $f, g \in C[x]$ of degree 2.

Output: All $\sigma \in M_C^{\text{type}}$ that map roots of f to roots of g.

1. Compute the roots of g in $C(\alpha) \cong C[x]/(f)$. (**Note:** there are either 0 or 2 roots β_j)

2. For each root β_j, compute $a, b, c \in C$ (not all 0) with $\frac{a\,\alpha + b}{c\,\alpha - a} = \beta_j$.
   This is done by computing coefficients (w.r.t α) of $a\,\alpha + b - \beta_j(c\,\alpha - a)$ and equating them to 0.

3. For each $\frac{a\,x + b}{c\,x - a}$ found in step 2 check if it fixes S_C^{type}, if so, include it in the output.

Algorithm: Case5

Input: S_C^{type} with S_i having ≥ 1 elements and $i \geq 3$.

Output: The set M_C^{type}.

Step 1: Find S_i for an $i \geq 3$ with $n_i > 0$.

Step 2: Choose a polynomial f in S_i. Denote $C(\alpha) \cong C[x]/(f)$, with $f(\alpha) = 0$.

Step 3: For each polynomial $g \in S_i$, call FindMaps(f, g). Then M_C^{type} would be $\bigcup_{g \in S_i} \text{FindMaps}(f, g)$.

4. COMPUTING 2-DESCENT, CASE A

Notations: Let $L \in C(x)[\partial]$ have order 2, and be irreducible (even in $\mathbb{C}(x)[\partial]$). Let $\sigma \in \text{Aut}(C(x)/C)$ have order 2 and fixed field $C(f) \subset C(x)$.

LEMMA 2. *If $\exists \tilde{L} \in \mathbb{C}(f)[\partial_f]$ with $L \sim_p \tilde{L}$, then $L \sim_p \sigma(L)$.*

PROOF. $L \sim_p \tilde{L} = \sigma(\tilde{L}) \sim_p \sigma(L)$. $\square$

So if not $L \sim_p \sigma(L)$ then $L \in C(x)[\partial] \subset \mathbb{C}(x)[\partial]$ does not descend to $\mathbb{C}(f)$. If $L \sim_p \sigma(L)$ then we will consider two cases:

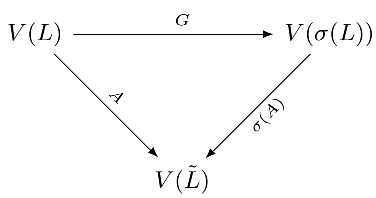
Notation 1. ***Case A*** *is when there exists* $G = r_0 + r_1\,\partial \in \mathbb{C}(x)[\partial]$ *such that* $G(V(L)) = V(\sigma(L))$*, i.e.* $L \sim_g \sigma(L)$*.*

Case B *is when there exists* $G = e^{\int r} \cdot (r_0 + r_1\,\partial)$ *such that* $G(V(L)) = V(\sigma(L))$*, i.e.* $L \sim_p \sigma(L)$*.*
(Note: *Case A* $\Rightarrow$ *Case B.)*

This section treats only Case A. Section 5 will reduce Case B to Case A.

In **Case A**, when $L \sim_g \sigma(L)$, it is known [11] that there exists $\tilde{L} \in \mathbb{C}(f)[\partial_f]$ with $\tilde{L} \sim_g L$. Then we have the following diagram:

Diagram 1

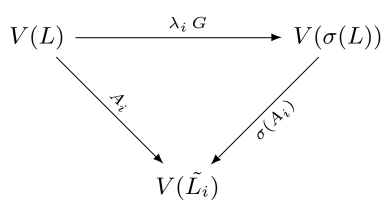


Here, A, $\sigma(A)$, and $\tilde{L}$ are unknown. Whether or not such a diagram commutes is studied in Theorem 1 below.

Remark 6. *A gauge transformation is a bijective map* $A : V(L) \to V(\tilde{L})$ *that can be represented by a differential operator in* $\mathbb{C}(x)[\partial]$*. So we can define* $\sigma(A)$ *simply by applying* σ *to the operator that represents the map* A*.*

Theorem 1. *Let* L *and* σ *be as before, and* $G : V(L) \to V(\sigma(L))$ *be a gauge transformation. Suppose* $\tilde{L_1}, \tilde{L_2} \in \mathbb{C}(f)[\partial_f]$ *and* $A_i : V(L) \to V(\tilde{L_i})$ *are gauge transformations. Then:*

1. *For each* $i = 1, 2$*, there is exactly one* $\lambda_i \in \mathbb{C}^*$ *such that the following diagram commutes.*
   Diagram 2



2. *If* $\tilde{L_1} \sim_g \tilde{L_2}$ *over* $\mathbb{C}(f)$*, then* $\lambda_1 = \lambda_2$*; Otherwise,* $\lambda_1 = -\lambda_2$*.*

3. *In particular,* $\{\lambda_1, -\lambda_1\}$ *depends only on* (L, σ, G)*.*

Proof. First consider the diagram without λ_i in it. In it we find two gauge transformations $V(L) \to V(\tilde{L_i})$, namely A_i and $\sigma(A_i)G$. After choosing bases of $V(L)$ and $V(\tilde{L_i})$, we can view these gauge transformations as bijections: $\mathbb{C}^2 \to \mathbb{C}^2$. Then by linear algebra, there is a constant $\lambda_i \in \mathbb{C}^*$ such that the map:

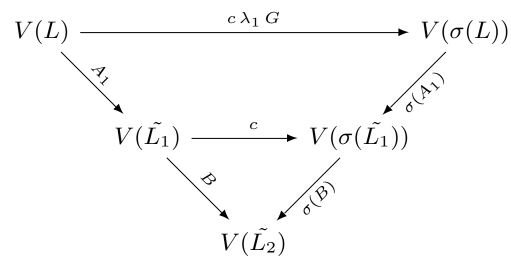
$$A_i - \lambda_i \sigma(A_i) G : \, V(L) \to V(\tilde{L_i}). \quad (4)$$

has a non-zero kernel. The kernel of (4) corresponds to a right hand factor of L, namely, the GCRD of L and the operator in (4). However, L is irreducible so this kernel must be $V(L)$ itself. That means Diagram 2 commutes. That λ_i is unique follows from linear algebra: there can be at most one λ_i for which (4) is the zero map. Item 1 follows.

For item 2, since $\tilde{L_1} \sim_g L \sim_g \tilde{L_2}$, there exists a gauge transformation $B : V(\tilde{L_1}) \to V(\tilde{L_2})$. This B is unique up to multiplying by a constant that we choose in such a way that the composition $BA_1 : V(L) \to V(\tilde{L_2})$ coincides with A_2. Since $\sigma(\tilde{L_1}) = \tilde{L_1}$, $\sigma(\tilde{L_2}) = \tilde{L_2}$ one sees that $\sigma(B)$ maps $V(\tilde{L_1})$ to $V(\tilde{L_2})$ as well. So $\sigma(B)$ must be $c \cdot B$ for some $c \in \mathbb{C}^*$. Then $|\sigma| = 2$ implies that $c = \pm 1$. Now $c = 1$ iff $\sigma(B) = B$ iff $B \in \mathbb{C}(f)[\partial_f]$ iff $\tilde{L_1}, \tilde{L_2}$ are gauge-equivalent over $\mathbb{C}(f)$. Otherwise, if $c = -1$, then $B \notin \mathbb{C}(f)[\partial_f]$ and $\tilde{L_1}, \tilde{L_2}$ are gauge-equivalent over $\mathbb{C}(x)$ but not over $\mathbb{C}(f)$. To prove item 2 we now have to show that $\lambda_2 = c\lambda_1$.

If λ_i is such that Diagram 2 commutes (for $i = 1, 2$) then the following diagram commutes:

Diagram 3



The composed map $B\,A_1$ at the left of Diagram 3 coincides with the map A_2 in Diagram 2 for $i = 2$. Applying σ to $B\,A_1$ and A_2, we see that the composed map at the right of Diagram 3 coincides with the map $\sigma(A_2)$ in Diagram 2 for $i = 2$. Then the maps at the top of Diagram 3 and Diagram 2 for $i = 2$ must coincide as well, i.e., $\lambda_2 G = c\lambda_1 G$. Hence $\lambda_2 = c\lambda_1$. Item 2 (and hence item 3) follow. □

4.1 Algorithm for finding 2-descent in Case A

Notations L, C, G, σ, A are as in Section 4. Our goal is to compute 2-descent: $L \sim_p \tilde{L} \in \mathbb{C}(f)[\partial_f]$. Here f is determined from σ as in Remark 2. We will compute $A : V(L) \to V(\tilde{L})$ first, then use A to find $\tilde{L}$.

Algorithm: Case A for computing a 2-descent $\tilde{L}$ for L.

Input: L, G, σ and C.

Output: $\tilde{L}$ and A, defined over an optimal extension of C.

Step 1: Write $A = (a_{00} + a_{01}x)\partial + (a_{10} + a_{11}x)$, with a_{00}, a_{01}, a_{10}, a_{11} unknowns (which will take values in $\mathbb{C}(f)$).

Step 2: The operator $A - \lambda\sigma(A)G$ in (4) should vanish on $V(L)$, so the remainder of $A - \sigma(A)\lambda G$ right divided by L must be 0. This remainder is of the form $(R_{00} + R_{01}x)\partial^0 + (R_{10} + R_{11}x)\partial$, where the R_{ij} are $C(\lambda, f)$-linear combinations of a_{ij}. This produces a system of 4 equations $R_{ij} = 0$ in 4 unknowns a_{ij}.

Step 3: To have a nontrivial solution, the corresponding 4×4 matrix M must have determinant 0. Equating $\det(M)$ to 0 gives a degree 4 equation for λ. Solve for λ.
Remark. The equation for λ is of the form $(\lambda^2 - a)^2 = 0$, where $a = \lambda_1^2 = \lambda_2^2$ with λ_1, λ_2 as in Theorem 1. If L and σ are defined over a field $C \subseteq \mathbb{C}$ then $\tilde{L}$ and A are defined over $C(\sqrt{a})$.
If $\sqrt{a} \notin C$ then it follows from Theorem 1 that the extension by $\lambda_i = \pm\sqrt{a}$ is necessary.

Step 4: Plug in one value for λ in M, then solve M to find values for $a_{00}, a_{01}, a_{10}, a_{11}$ in $C(\sqrt{a}, f)$.

Step 5: Compute LCLM(A, L) to obtain $\tilde{L}A$. Right divide by A to find $\tilde{L} \in C(\sqrt{a}, f)[\partial_f]$.

Step 6: (optional) Introduce a new variable, say x_1, and compute an operator $L_{x_1} \in C(\sqrt{a}, x_1)[\partial_{x_1}]$ that corresponds to $\tilde{L}$ under the change of variables $x_1 \mapsto f$.

5. COMPUTING 2-DESCENT, CASE B

DEFINITION 7. *Let L_1, $L_2 \in \mathcal{D} = K[\partial]$. The* symmetric product $L_1 Ⓢ L_2$ *is defined as the monic differential operator in $\mathcal{D}$ with minimal order for which $y_1\, y_2 \in V(L_1 Ⓢ L_2)$ for all $y_1 \in V(L_1)$, $y_2 \in V(L_2)$.*

LEMMA 3. *If $L = \partial^2 + c_0 \in C(x)[\partial]$, and $G := e^{\int r} \cdot (r_0 + r_1\partial)$ is a bijection from $V(L)$ to $V(\sigma(L))$, then $(e^{\int r})^2$ is a rational function.*
If $L := \partial^2 + a_1\partial + a_0 \in \mathbb{C}(x)[\partial]$, then $L_1 := L Ⓢ (\partial - \frac{1}{2}a_1)$ is of the form $\partial^2 + c_0$ (with $c_0 = a_0 - \frac{1}{4}a_1^2 - \frac{1}{2}a_1')$.

The proof of the lemma follows by computing the effect of G on the Wronskian, and the fact that the Wronskians of $\partial^2 + c_0$ and $\sigma(\partial^2 + c_0)$ are rational functions (1 and $\sigma(x)'$ respectively).

Let $L \in C(x)[\partial]$ irreducible (even over $\mathbb{C}$) and of order 2, and $\sigma \in \mathrm{Aut}(C(x)/C)$ of order 2. The implementation equiv [8] can check if $L \sim_p \sigma(L)$, and if so, find $r, r_0, r_1 \in C(x)$ for which $G := e^{\int r} \cdot (r_0 + r_1\partial)$ is a bijection from $V(L)$ to $V(\sigma(L))$. Assume that such σ and G are given. After the simple transformation in the lemma above, we may assume that $(e^{\int r})^2$ is a rational function.

If $e^{\int r}$ itself is a rational function, then we are in Case A. Otherwise, we can write $e^{\int r} = p(x)\sqrt{f(x)}$ for some square-free polynomial $f(x)$, and some $p(x) \in C(x)$.

DEFINITION 8. *The branch points of G are the roots of $f(x)$, and ∞ if $f(x)$ has odd degree.*

To reduce Case B to Case A, we have to eliminate the branch points. Our algorithm will first eliminate all branch points that can be eliminated without a field extension of C. It will only extend C if there is no descent w.r.t. σ defined over C.

5.1 Branch points

It is convenient to view the set of branch points as a subset of $\mathbb{P}^1(\overline{C})$. However, to avoid splitting fields, the algorithm represents the branch points with a set $B \subset \mathrm{places}(C)$ instead. This B is the set of irreducible factors of $f(x)$ in $C[x]$, as well as ∞ if $f(x)$ has odd degree. The goal is to eliminate branch points until we reach $B = \emptyset$, i.e., Case A.

DEFINITION 9. *If $\sigma(\infty) = \infty$, then denote* $\mathrm{Inf} := \{\infty\}$, *otherwise* $\mathrm{Inf} := \{\infty, x - \sigma(\infty)\}$. *Denote $B_I = B \bigcap \mathrm{Inf}$ and $B_N = B \setminus B_I$.*
Let $f_1(x), f_2(x) \in B_N$. We say that $f_1(x)$ matches $f_2(x)$ when the roots of $f_2(x)$ are the same as the roots of $f_1(\sigma(x))$ (i.e. the numerator of $f_1(\sigma(x))$ is f_2).
If $\sigma(\infty) \neq \infty$, then we say that the polynomial $x - \sigma(\infty)$ matches ∞.

LEMMA 4. *If $f_1(x) \neq f_2(x) \in B_N$ and $f_1(x)$ matches $f_2(x)$, then B_N turns into $B_N \setminus \{f_1, f_2\}$ when we replace L by $L_{\mathrm{new}} := L Ⓢ (\partial - \frac{1}{2} \cdot \frac{f_1(x)'}{f_1(x)})$.*

PROOF. The composed transformation

$$V(L_{\mathrm{new}}) \to V(L) \to V(\sigma(L)) \to V(\sigma(L_{\mathrm{new}}))$$

is

$$\sqrt{\sigma(f_1)} \cdot G \cdot \frac{1}{\sqrt{f_1}}.$$

The polynomial f equals $f_1 f_2 \cdots$ where the $\cdots$ refer to the other factors of f in $B \setminus \{\infty\}$. The transformation G is of the form $\sqrt{f_1 f_2 \cdots} \cdot (r_0 + r_1\partial)$. Factors can be removed from the square-root in G either by division or by multiplication by a square-root (factors in $C(x)$ can be moved to r_0, r_1). So in the composed transformation, the factors f_1 and f_2 will disappear from the square-root in G (note: this uses the assumption $f_1 \neq f_2$ (which implies that their gcd is 1 since they are monic irreducible polynomials)).
A subtlety is that if $\sigma(\infty) \neq \infty$, then $\sigma(f_1)$ is not f_2 but $cf_2/(x - \sigma(\infty))^d$, for some $c \in C$, where d is the degree of f_1 and f_2. This means that if $\sigma(\infty) \neq \infty$ and d is odd, then the set B_I will change when we replace L by L_{new} ($B_I = \emptyset$ will change to Inf, and $B_I = \mathrm{Inf}$ will change to $\emptyset$). $\square$

LEMMA 5. *If $\sigma(\infty) \neq \infty$, and $B_I = \{\infty, f_1\}$ (here $f_1 = x - \sigma(\infty)$) then the factor f_1 inside the square root in G will cancel out (i.e. B_I will become $\emptyset$) if we replace L by $L_{\mathrm{new}} := L Ⓢ (\partial - \frac{1}{4} \cdot \frac{1}{f_1})$.*

PROOF. The solutions of L_{new} differ a factor $\sqrt[4]{f_1}$ from the solutions of L. The lemma follows from a similar computation as the proof of Lemma 4, except that this time $\sigma(f_1)$ is of the form c/f_1 for some constant c. Thus, the composed map is of the form $\sqrt[4]{c/f_1} \cdot G \cdot 1/\sqrt[4]{f_1}$, and $\sqrt{f_1}$ is cancelled from the square root in G. $\square$

In the following algorithm, L and σ are as in Section 4, and $G = e^{\int r} \cdot (r_0 + r_1\partial)$ with $r, r_0, r_1 \in C(x)$.

Algorithm: Case B for computing a 2-descent $\tilde{L}$ for L.

Input: L, G, σ and C.

Output: $\tilde{L}$ and A (defined over C whenever possible).

Step 1 Initialization: If $(e^{\int r})^2$ is not a rational function, then replace L by $L Ⓢ (\partial - \frac{1}{2} \cdot \frac{a_1}{a_2})$ as in Lemma 3 and update G accordingly.
Rewrite G as $\sqrt{f(x)}(r_0 + r_1\partial)$ with $f(x)$ monic and square-free (updating $r_0, r_1 \in C(x)$ to move any rational factor from $e^{\int r}$ to r_0, r_1).
If $f(x) = 1$ then call **Case A** and stop.

Step 2: Factor $f(x)$ in $C[x]$ to find $B, B_I, B_N \subset \mathrm{places}(C)$.

Step 3: $g :=$**Findg**(B_N, σ, C).
(See below for the subalgorithm **Findg**)

Step 4: Let $h := \frac{1}{2} \cdot \frac{g'}{g}$. Replace L by $L Ⓢ (\partial - h)$ and update G, B, B_I, B_N accordingly. Now B_N should be $\emptyset$.

Step 5: If $B_I \neq \emptyset$ then let $h := \frac{1}{4} \cdot \frac{1}{f_1}$ with f_1 as in Lemma 5. Replace L by $L Ⓢ (\partial - h)$ and update G, B accordingly. Now B should be $\emptyset$.

Step 6: Call **Case A**.

Subalgorithm: Findg.

Input: B_N, σ, C.

Output: g.

Step 1: If $B_N = \emptyset$, return 1 and stop.

Step 2: Else, for each $P_i \in B_N$,

1. Find its matched (Def. 9) element $P_j \in B_N$.
2. If $P_i \neq P_j$ then $g := \mathbf{Findg}(B_N \setminus \{P_i, P_j\}, \sigma, C)$, return $g \cdot P_i$ and stop.

Step 3: Now each $P \in B_N$ matches itself, and hence has even degree. Choose $P \in B_N$ with minimal degree, and let $\alpha \in \overline{C}$ be one root of P, so $C(\alpha) \cong C[x]/(P)$. Let B_N^α be the set of all irreducible factors in $C(\alpha)[x]$ of all elements of B_N. Return $\mathbf{Findg}(B_N^\alpha, \sigma, C(\alpha))$.

6. MAIN ALGORITHM

Algorithm 2-descent.

Input: A second order irreducible differential operator $L \in C(x)[\partial]$ and the field C.

Output: descent, if it exists for some $\sigma \in \mathrm{Aut}(C(x)/C)$ of order 2.

Step 1: Compute the set of true singularities, and the singularity structure S_C^{type}.

Step 2: Call **Compute Möbius transformations** in Section 3.2 to compute the set M_C^{type}.

Step 3: For each $\sigma \in M_C^{\mathrm{type}}$, call [8] to check if $L \sim_p \sigma(L)$, and if so, to find $G : V(L) \to V(\sigma(L))$.
If we find σ with $L \sim_p \sigma(L)$, then call algorithm Case B in Section 5.1 and stop.

7. EXAMPLES

We give two examples. The first example is easy (it has $G = r_0 + r_1\partial$ with $r_1 = 0$). The second one is less trivial[3]. The first example is in **Case A** as in Section 4, the second example involves both **Case A** and **Case B**.

Example 2. *Let*

$$L = \partial^2 + \frac{28x-5}{x(4x-1)}\partial + \frac{144x^2+20x-3}{x^2(4x-1)(4x+1)}$$

Step 1: *Compute the singularity structure of L*

$$S_C^{\mathrm{type}} := \{(x,0), (\infty,0), (x-\frac{1}{4},0), (x+\frac{1}{4},0)\}$$

Step 2: *Compute Möbius transformations. Since S_1 has $n_1 = 4$ elements, we end up in algorithm **Case1** of Section 3.2 which produces:*

$$\{-x, \frac{-1}{16x}, \frac{1}{16x}, \frac{-1}{4}\frac{4x-1}{4x+1}, \frac{1}{4}\frac{4x+1}{4x-1}\}$$

Step 3: *There are 5 choices for σ. The first one is $x \mapsto -x$ corresponding to the subfield $C(f) = C(x^2)$. The **equiv** [8] program finds $G = \frac{4x-1}{4x+1}$. Next we compute $A := -4x^2 + x$, and then $\tilde{L}$. After applying a change of variable $x \mapsto \sqrt{x_1}$ the result reads*

$$L_{x_1} := (16x_1 - 1)x_1\partial^2 + (32x_1 - 2)\partial + 4$$

which has 3 true singularities and is easy to solve.

[3]it was e-mailed to one of us to find its closed form solutions. There have been many such requests, which motivates us to develop these algorithms.

Example 3. *Consider the operator:*

$$L := \partial^2 + \frac{4(1296x^5+576x^4-144x^3-72x^2+x+1)}{x(6x-1)(2x+1)(6x+1)(12x^2-1)}\partial + \frac{2(5184x^6-864x^5-1656x^4+48x^3+162x^2+6x-1)}{(-1+2x)x^2(6x-1)(2x+1)(6x+1)(12x^2-1)}$$

Step 1: *Compute the singularity structure of L*

$$S_C^{type} := \{(x,0), (\infty,0), (x-\frac{1}{2},0), (x+\frac{1}{2},0), (x-\frac{1}{6},0), (x+\frac{1}{6},0)\}$$

($12x^2 - 1$ is a removable singularity, Definition 5).

Step 2: *Compute Möbius transformations. Since S_1 has $n_1 = 6$ elements, we are again in Case1, and find:*

$$\{-x, \frac{-1}{12x}, \frac{1}{12x}, \frac{-1}{2}\frac{2x-1}{6x+1}, \frac{1}{2}\frac{2x+1}{6x-1}, \frac{-1}{6}\frac{6x-1}{2x+1}, \frac{1}{6}\frac{6x+1}{2x-1}\}$$

Step 3: *The first σ we try is $x \mapsto -x$. The **equiv** program finds*

$$G := \frac{x(12x^2+4x-1)}{12x^2-1}\partial + \frac{3}{2}\frac{(2x+1)(10x-1)}{12x^2-1}$$

so $G(V(L)) = V(\sigma(L))$. Then compute a 4 by 4 matrix from the linear equations for the a_{ij}, equate the determinant to 0 and find $\lambda = \pm 2$. We choose $\lambda = 2$ and find

$$A := (-36x^4 - \frac{1}{4} + 10x^2)\partial + 1 - \frac{1}{4}\frac{(288x^4+1-84x^2)}{x}.$$

We get

$$\begin{aligned} L_{x_1} :=& 4x_1^2(-1+36x_1)(4x_1-1)(12x_1-1)^2\partial^2 + \\ & 8x_1(12x_1-1)(4x_1-1)(216x_1^2-54x_1+1)\partial \\ & -3 - 2544x_1^2 + 10368x_1^3 + 48x_1 \end{aligned}$$

*which is $\tilde{L} \in C(x^2)[\partial_{x^2}]$ rewritten with $x \mapsto \sqrt{x_1}$. This L_{x_1} has 4 true singularities, and allows a further 2-descent. Applying steps (1)(2)(3) to L_{x_1} again, we are actually in **Case B** as in Section 5, applying the algorithm (details are given in a Maple worksheet [5]) we find a new operator $\tilde{L_1} \sim_p L_{x_1}$ defined over the subfield $\mathbb{C}(f_1)$ where $f_1 := x_1 + \frac{1}{144x_1}$. Replacing f_1 by a new variable x_2 we get:*

$$\begin{aligned} L_{x_2} :=& 4(36x_2+11)(18x_2-5)(6x_2+1)(6x_2-1)^2\partial^2 + \\ & 36(6x_2-1)(1296x_2^3+1620x_2^2+20x_2-9)\partial \\ & + 34992x_2^3 - 207036x_2^2 - 2331 + 3456x_2 \end{aligned}$$

which has 3 true regular singularities (as well as a few removable singularities). That means that L_{x_2} (and hence L) has closed form solutions (see [5]) in terms of hypergeometric ${}_2F_1$ functions.

8. FUTURE WORK

At the moment, we only consider σ's that are defined over the same field of constants C over which L is defined. We can modify the Compute Möbius transformations algorithm to also find σ's defined over an extension of C. However, for such σ we do not plan to compute 2-descent because if there exists descent w.r.t. a σ that is not defined over C, then a larger descent should exist as well.

We plan to work on finding (if it exists) descent to subfields of index 3. Degree 3 extensions need not be Galois, and so in general, to find 3-descent it is not enough to try all Möbius transformations that fix the singularity structure.

9. REFERENCES

[1] Barkatou, M. A., and Pflügel, E. On the Equivalence Problem of Linear Differential Systems and its Application for Factoring Completely Reducible Systems. In *ISSAC 1998*, 268–275.

[2] Bronstein, M. An improved algorithm for factoring linear ordinary differential operators. In *ISSAC 1994*, 336–340.

[3] Compoint, E.,van der Put, M.,and Weil, J.A. Effective descent for differential operators. *J. Algebra. 324* (2010), 146–158.

[4] Debeerst, R, van Hoeij, M, and Koepf. W. Solving Differential Equations in Terms of Bessel Functions. In *ISSAC 2008*, 39–46.

[5] Fang, T. Implementation and examples for 2-descent www.math.fsu.edu/∼tfang/2descentprogram/

[6] van Hoeij, M., and Vidunas, R. All non-Liouvillian ${}_2F_1$-solvable Heun equations with pullbacks in $\mathbb{C}(x)$. www.math.fsu.edu/∼hoeij/files/Heun/TextFormat/

[7] van Hoeij, M. *Factorization of Linear Differential Operators*. PhD thesis, Universiteit Nijmegen, 1996.

[8] van Hoeij, M. Implementation for finding equivalence map. *www.math.fsu.edu/∼hoeij/files/equiv.*

[9] van Hoeij, M. Solving Third Order Linear Differential Equations in Terms of Second Order Equations. In *ISSAC 2007*, 355–360. Implementation at: *www.math.fsu.edu/∼hoeij/files/ReduceOrder*

[10] van Hoeij, M, and Cremona, J. Solving conics over function fields. *J. de Theories des Nombres de Bordeaux.18* (2006), 595–606.

[11] van Hoeij, M, and van der Put, M. Descent for differential modules and skew fields. *J. Algebra. 296* (2006), 18–55.

[12] van Hoeij, M, and Yuan, Q Finding all Bessel type solutions for Linear Differential Equations with Rational Function Coefficients. In *ISSAC 2010*, 37–44

[13] van der Hoeven, J. Around the Numeric-Symbolic Computation of Differential Galois Groups. *J. Symb. Comp. 42* (2007), 236–264.

[14] Nguyen, A. K. A modern perspective on Fano's approach to linear differential equations. PhD thesis (2008).

[15] van der Put, M., and Singer, M. F. *Galois Theory of Linear Differential Equations*, vol. 328 of *A Series of Comprehensive Studies in Mathematics.* Springer, Berlin, 2003.